HPSS
User's Guide

High Performance Storage System
Release 4.1.1

November 1999 (Revision 2)

HPSS User's Guide

Portions of this work were produced by the University of California,Lawrence Livermore National Laboratory (LLNL) under Contract No. W-7405-ENG-48 with the U.S. Department of Energy (DOE), by the University of California, Lawrence Berkeley National Laboratory (LBNL) under Contract No. DEAC03776SF00098 with DOE, by the University of California, Los Alamos National Laboratory (LANL) under Contract No. W-7405-ENG-36 with DOE, by Sandia Corporation, Sandia National Laboratories (SNL) under Contract No. DEAC0494AL85000 with DOE, and Lockheed Martin Energy Research Corporation, Oak Ridge National Laboratory (ORNL) under Contract No. DE-AC05-96OR22464 with DOE. The U.S. Government has certain reserved rights under its prime contracts with the Laboratories.

DISCLAIMER

Portions of this software were sponsored by an agency of the United States Government. Neither the United States, DOE, The Regents of the University of California, Sandia Corporation, Lockheed Martin Energy Research Corporation, nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

Printed in the United States of America.

HPSS Release 4.1.1

November 1999 (Revision 2)

Preface

This High Performance Storage System (HPSS) User's Guide provides the necessary information for transferring files using HPSS. It is designed for HPSS users.

The document provides the User's Guide information for HPSS Release 4.1.1. In particular, the following interfaces are described: standard File Transfer Protocol (FTP) interface, Parallel FTP (PFTP) interface, Network File System Version 2 (NFS V2), and Distributed File System (DFS). **Note:** It is not the intent of this document to define the standard commands and subcommands provided by standard FTP, NFS, and DFS. Only interface extensions provided by HPSS are defined within the HPSS User's Guide.

Refer to the HPSS Administration Guide for a description of the interfaces provided to HPSS administrators. Programming interfaces are documented in the HPSS Programmer's Reference Guides.

Refer to the HPSS Programmer's Reference Guide, Volume 1 for programming interfaces provided to the end user. The programming interfaces provided in the Programmer's Guide Volume 1 are: HPSS Client Application Programming Interface (API), and 64 bit arithmetic functions.

Refer to the HPSS Error Messages Manual for a list of all HPSS error and advisory messages which are output by the HPSS software. For each message, the following information is provided: message identifier and text, source file name(s) which generated the message, problem description, system action, and administrator action.

Refer to the HPSS Programmer's Reference Guide, Volume 2 for programming interfaces to each HPSS server. While it is envisioned that most users will access HPSS through the client API, standard FTP, PFTP, NFS, or the DFS interfaces (documented in Volume 1), well defined programming interfaces are defined for each HPSS server. It should be noted that programming to the individual server level is a more complex programming model which requires a greater understanding of the HPSS servers.

The HPSS User's Guide is structured as follows:

Chapter 1: Overview	This chapter provides an overview of each type of user interface, a summary of key storage concepts, and recommendations on usage.
Chapter 2: File Transfer Protocol (FTP)	This chapter defines the extensions to the standard FTP interface.
Chapter 3: Parallel File Transfer Protocol (PFTP)	This chapter defines the Parallel FTP (PFTP) inter-

	face.
Chapter 4: User Utilities	This chapter defines the set of utilities available to the general user.
Appendix A: Acronyms	This appendix A provides a list of acronyms used in this document.
Appendix B: References	This appendix lists documents cited in the text as well as other reference materials.

Typographic and Keying Conventions

This document uses the following typographic conventions:

Bold	**Bold** words or characters represent system elements that you must use literally, such as functions, commands or keywords.
Italic	*Italic* words or characters represent variable values to be supplied.
[ ]	Brackets enclose optional items in syntax and format descriptions.
{ }	Braces enclose a list of items to select in syntax and format descriptions.

Table of Contents

Chapter 1. Overview

The High Performance Storage System (HPSS) provides scalable parallel storage systems for highly parallel computers as well as traditional supercomputers and workstation clusters. Concentrating on meeting the high end of storage system and data management requirements, HPSS is scalable and designed for large storage capacities, and to use network-connected storage devices to transfer data at rates up to multiple gigabytes per second. Listed below are the user interfaces for accessing data from HPSS.

1.1. User Interfaces

1.1.1. File Transfer Protocol (FTP)

HPSS provides an industry-standard FTP user interface. Because FTP is a serial interface, data sent to a user is received serially. This does not mean that the data within HPSS is not stored and retrieved in parallel. It simply means that the FTP daemon within HPSS must consolidate its internal parallel transfers into a serial data transfer to the user. HPSS FTP performance in many cases will be limited not by the speed of a single storage device, as in most other storage systems, but by the speed of the data path between the HPSS FTP daemon and the user's FTP client.

All FTP commands are supported or properly rejected if the HPSS Parallel FTP Daemon does not implement a specific feature. In addition, the ability to specify Class of Service for is provided via the **quote site** or **site** commands. Additional site command options are provided for chgrp, chgid, chmod, chown, chuid. stage, wait, and symlink. The HPSS FTP Daemon supports access from any RFC-0959 conformant FTP Client. In addition, the **quote allo64** command is supported.

Passive connections are not supported. Also, to avoid confusion, the user may want to explicitly specify the data transfer type of ascii or binary.

Refer to the HPSS System Administration Guide for information on configuring PFTP.

1.1.2. Parallel FTP (PFTP)

The PFTP supports normal FTP plus extensions. It is built to optimize FTP performance for storing and retrieving files from HPSS by allowing the data to be transferred in parallel to the client. The interface pro-

vided to the user has syntax similar to FTP but with some extensions to allow the user to transfer data to and from HPSS across parallel communication interfaces. PFTP supports transfers either via TCP/IP or IPI-3 over HIPPI. In either case, the FTP client communicates directly with HPSS Movers to transfer data.

The following constraints are imposed by PFTP.

- Pipes are not supported.
- Passive connections are not supported.
- ASCII transfers are not supported over the parallel interface because ASCII transfers insert characters. This makes it impossible to send the data in parallel. Since extra characters are inserted in the stream, there is no way to resolve data placement. Warning: Some FTP implementations default to ascii. If this is the case, it will be necessary to specify binary by entering the bin command.
- PFTP client access is supported only from nodes which support the HPSS PFTP client software.

Refer to the HPSS System Administration Guide for information on configuring PFTP.

1.1.3. Network File System Version 2 (NFS V2)

The purpose of the NFS V2 server interface is to provide transparent access to HPSS name space object and bitfile data for client systems. Following a mount on the HPSS file system name, the user may access HPSS files using standard function calls and command interfaces.

The code written to implement the NFS V2 Server interface is written to the *Network File System Specification, RFC-1094*, DDN Network Information Center, SRI International, Menlo Park, Ca.

The NFS V2 Server interface and data structures are defined by RFC-1094.

Since there are no extensions or modifications to the NFS user interface, no additional interface information is provided in the remainder of this document.

The following constraints are imposed by the HPSS NFS V2 server.

- 2 gigabyte maximum file size. This limitation is imposed by NFS - not HPSS.
- All files created using NFS are stored in a single Class of Service. The Class of Service used by NFS is defined by the HPSS administrator.

- NFS transfers are slower than the other HPSS interfaces, and are therefore not recommended for large file accesses.

Refer to the HPSS Administration Guide for information on configuring NFS V2.

1.1.4. Distributed File System (DFS)

HPSS has an option that enables it to interface with Transarc's DFS to provide distributed file system services. If a site opts to use DFS, the standard DFS interfaces can be used to administer filesets[1] and access the objects within the DFS filesets. Additional HPSS administrative set-up is required for HPSS to "know" about DFS filesets.

In addition to providing a distributed file system, high-speed I/O performance is essential to many sites. Since DFS could not be modified, without significant cost, to provide a high-speed I/O interface, two HPSS data management configuration options, at the fileset level, are available. Although these options are defined per fileset, it must be noted that because of limitations in the implementation, **all** filesets on a particular aggregate[2] must be configured with the same option.

Archived Filesets:
If the objects in a fileset will only be accessed through DFS interfaces, the fileset should be managed with the archived fileset option. HPSS is used strictly as an archive facility for DFS, with this option. Any DFS file data that has not been accessed recently will be migrated to HPSS, purged from DFS, and restored to DFS upon future access. At any given time, the data in a DFS file may be out of date with the data in the HPSS file. Clients may not use the HPSS interface to access objects in archived filesets. This option is well suited to applications that run directly on DFS, require more disk space than is available to DFS, and do not need a high-speed I/O interface.

Archived filesets incur some overhead when files are created and deleted. The overhead occurs because some processing is required to set state indicating a new file exists that must eventually be migrated to HPSS. When a file is unlinked some processing is also required to determine if the file must be marked for subsequent removal from HPSS. Removing the file from HPSS will take place at a later time. Delaying the file removal from HPSS allows for the file to be restored to DFS, if so desired.

Mirrored Filesets:
If the high-speed I/O interface to file data is required, the fileset must be managed with the mirrored fileset option. Typically, such applications would write large amounts of data to a file through the HPSS

1. Filesets are an administrative entity defined by DFS. They are simply a collection of files, or a directory subtree.
2. Aggregate is a term defined by DFS, and refers to an allocation of disk space.

interface, and later, read smaller pieces of this data through DFS. With this option, consistency between the HPSS and DFS name and data spaces is maintained. Name or data space updates made through the DFS interface will be visible through the HPSS interface, and vice versa. A user may access data through DFS, at typical DFS rates, and when high performance I/O rates are important, use the HPSS high-speed interface.

Mirrored filesets incur additional overhead for any name space activity that alters the name space, such as, creates, unlinks, renames, and owner or permissions changes. Name space activity that does not alter the name space, such as, “ls” and “cd”, perform at typical DFS rates.

Usage Considerations:
The overhead associated with DFS client I/O to files is insignificant for both mirrored and archived filesets, unless the data must be cached from HPSS to DFS. Moving data between DFS and HPSS is highly dependent on disk speed, network speed, and system load, but, in general, our data transfers between DFS and HPSS were as fast as permitted by the hardware.

Both archived and mirrored filesets support whole or partial file caching of data from HPSS to DFS. The caching option applies to **all** filesets on an aggregate.

It is critical to understand the differences in the HPSS functionality that is provided by archived and mirrored filesets. For example, with mirrored filesets DFS and HPSS name spaces are identical, and any name space changes made through either interface are immediately visible through both the DFS and HPSS interfaces. With archived filesets, there is no correspondence between the DFS and HPSS name space. It is important to configure the fileset type to suit the behavior expected of the files in the fileset.

Since there are no extensions or modifications to the DFS user interface, no additional interface informatiRefer to the HPSS Administration Guide for information on configuring DFS.

1.1.5. User Utilities

The purpose of the HPSS user utilities is to provide the end user with information such as Access Control List (ACL) definitions and Class of Service definitions. In addition, the ability for a user to change his ACL definitions is provided.

The user utilities consist of these commands:

- **chacl** change an ACL

- **lsacl** list an ACL
- **lshpss** list information for HPSS (Class of Service list, hierarchy list, storage class list, physical volumes, devices and drives, servers, Movers, and metadata)

1.2. Storage Concepts

This section defines key HPSS storage concepts which have a significant impact on the usability of HPSS. Configuration of the HPSS storage objects and policies is the responsibility of your HPSS administrator.

1.2.1. Class of Service

Class of Service (COS) is an abstraction of storage system characteristics that allows HPSS users to select a particular type of service based on performance, space, and functionality requirements. Each COS describes a desired service in terms of characteristics such as minimum and maximum file size, transfer rate, access frequency, latency, and valid read or write operations. A file resides in a particular COS and the class is selected when the file is created. Underlying a COS is a storage hierarchy that describes how data for files in that class are to be stored in HPSS.

For the FTP and PFTP interfaces, the COS ID may be explicitly specified by using the **site setcos** command. If not specified, a default COS is used. For NFS, all files are created in the same COS. This COS is defined by your system administrator. For DFS, COS may be assigned by explicitly assigning a COS in the fileset definition. Otherwise, the size of the file is used as hints for COS selection. Contact your HPSS administrator to determine the COSs which have been defined. The **lshpss -cos** command may also be used to list the defined COSs. Refer to Chapter 5 for information on the **lshpss** command.

Also, PFTP provides a feature to automatically store the local file size in the minimum and maximum file size fields of the COS. This feature is also provided for FTP clients which support the ALLO command. This allows the COS selection to be made according to file size. The HPSS administrator should ensure that COS definitions contain proper minimum and maximum file sizes in order for PFTP (FTP clients which support ALLO) to optimize storage utilization when tranferring files to HPSS. Note: If the COS ID is explicitly set by using the **site setcos** command, that COS will be used regardless of file size.

A COS is implemented by a Storage Hierarchy of one to many Storage Classes. Storage Hierarchies and Storage Classes are not directly visible to the user, but are described below since they map to COS. The relationship between storage class, storage hierarchy, and COS is shown in Figure 1-1.

1.2.2. Storage Class

An HPSS Storage Class is used to group storage media together to provide storage with specific characteristics for HPSS data. The attributes associated with a storage class are both physical and logical. Physical media in HPSS are called physical volumes. Physical characteristics associated with physical volumes are the media type, block size, the estimated amount of space on volumes in this class, and how often to write tape marks on the volume (for tape only). Physical media are organized into logical virtual volumes. This allows striping of physical volumes. Some of the logical attributes associated with the storage class are virtual volume block size, stripe width, data transfer rate, latency associated with devices supporting the physical media in this class, and storage segment size (disk only). In addition, the storage class has attributes that associate it with a particular migration policy and purge policy to help in managing the total space in the storage class.

1.2.3. Storage Hierarchy

An HPSS storage hierarchy consists of multiple levels of storage with each level representing a different storage media (i.e., a storage class). Files are moved up and down the storage hierarchy via stage and migrate operations, respectively, based upon storage policy, usage patterns, storage availability, and user request. For example, a storage hierarchy might consist of a fast disk, followed by a fast data transfer and medium storage capacity robot tape system, which in turn is followed by a large data storage capacity, but relatively slow data transfer tape robot system. Files are placed on a particular level in the hierarchy depending on the migration policy and staging operations. Multiple copies of a file may also be specified in the migration policy. If data is duplicated for a file at multiple levels in the hierarchy, the more recent data is at the higher level (lowest level number) in the hierarchy. Each hierarchy level is associated with a single storage class.

1.2.4. File Family.

A file family is an attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes. Release 4.1 supports grouping of files only on tape volumes. In addition, families can only be specified in Release 4.1 by associating a family with a fileset, and creating the file in the fileset. When a file is migrated from disk to tape, it is migrated to a tape virtual volume assigned to the family associated with the file. If no family is associated with the file, the file is migrated to the next available tape not associated with a family (actually to a tape associated with family zero). If no tape virtual volume is associated with the family, a blank tape is reassigned from family zero to the file's family. The family affiliation is preserved when tapes are repacked. Configuring file families is a System Administrator function.

1.3. Interface Usage Considerations

Guidance on when to use a particular HPSS interface is provided below. In general, PFTP provides the best data transfer performance. NFS is the slowest interface, and should not be the interface of choice for large HPSS data transfers.

Conditions in which the user might elect to use FTP are:

- Utilizes standard FTP interface

  Users and applications familiar with FTP can access HPSS with the standard command set.

- Supports files greater than 2 gigabytes

  FTP supports file sizes up to 2**64.

- Supports any FTP client platforms

  FTP commands may be issued from any vendor nodes with an FTP interface. No specialized code is required.

Conditions in which a user might elect to use PFTP are:

- Provides faster file transfers.

  PFTP is a better performer than FTP since it provides the capability to stripe data across multiple client data ports. In addition, IPI-3 transfers to the client are also supported.

- Supports files greater than 2 gigabytes

  PFTP supports file sizes up to 2**64.

- Supports partial file transfer.

  PFTP provides options on the pget and pput commands to perform partial file transfers. This would be beneficial to users who want to extract pieces of large files.

Conditions in which a user might elect to use NFS are:

- Provides standard system access

  Files may be accessed and managed through standard system mechanisms without calling a special library or program to translate commands

- Eliminates multiple file instances

  The need to maintain multiple instances of a file can be eliminated since files remain on the NFS server.

- Accesses limited to smaller files

  NFS V2 is limited to 2 gigabyte file sizes, and data transfer performance is slower than the other interfaces.

- Supports any NFS client platforms

  NFS access is supported from any vendor nodes with an NFS V2 interface. No specialized code is required.

Conditions in which a user might elect to use DFS are:

- Provides standard system interface

  Files may be accessed and managed through standard system mechanisms without calling a special library or program to translate commands

- Supports any DFS client platforms

  DFS access is supported from any vendor nodes with aDFS client interface.

- Supports files larger than 2 gigabytes.

- Small file performance.

  Provide DFS transfer rates for smaller files (unless migrated to HPSS)

- Other

  DFS provides security and global name space features

  Extend DFS storage through a back-end to mass storage.

  Provide mirrored option which supports data accesses from either the DFS interface or any of the HPSS interfaces. This would be used for large files with higher performance demands.

1.4. User IDs

After the HPSS system is configured, the necessary accounts must be created for HPSS users. Contact your HPSS administrator to add an account.

For FTP / PFTP access, an FTP account must be created. The administrator can use the **hpssuser -add** *user* **-ftp** command to add a new FTP user.

For NFS access, a DCE user account must be created. The administrator can use the **hpssuser -add** *user* **-dce** command to add a new DCE account.

For DFS access, the user must have a DCE account and be logged in to DCE.

Users calling the utilities described in this document must be logged into DCE. As noted above, the administrator can use the **hpssuser -add** *user* **-dce** command to add a new DCE account.

1.5. DCE User Accounts

As mentioned in the previous section, the user utilities and DFS require the user be logged into DCE.

The following command is used to issue a DCE login:

dce_login [*principal_name*] [*password*]

When this command is entered, the principal's identity is validated, and the network credentials are obtained. If *principal name* or *password* are not supplied, **dce_login** will prompt for them.

When the principal's DCE login context is no longer required, the following command may be used to destroy the login context and associated credentials:

kdestroy

Other DCE commands which might be of interest to the user are:

klist list the primary principal and tickets held in the DCE credentials cache

kinit Refresh a DCE credentials cache

Chapter 2. File Transfer Protocol (FTP)

This chapter specifies the HPSS FTP interface. For information on configuring the HPSS FTP daemon, reference the *HPSS Administration Guide*. FTP is supported from any FTP client platform.

HPSS supports the FTP command set for transferring files to and from HPSS. To use FTP, the user enters the following:

ftp *node_name* [*port_number*]

where,

node_name is the node name of the node where the HPSS FTP Daemon process resides

port_number is the port number for HPSS, as set up in /etc/services

At this point, any standard FTP command may be entered. Note: If the message "Load thread state failed" is received, contact your HPSS administrator. This message generally implies that either HPSS is not correctly configured, or some HPSS components may not be executing:

2.1. Site Commands

HPSS also supports the site commands listed below. Note: On some platforms, it may be necessary to specify **quote site** instead of **site**.

- **setcos**
- **chgid**
- **chgrp**
- **chmod**
- **chown**
- **chuid**
- **stage**

- **wait**
- **symlink**

2.1.1. Specifying a File's Class of Service - setcos

setcos is used to specify a class of service and has the following format:

```
site setcos cos_id
```

where,

cos_id is the Class of Service identifier (used when creating a new HPSS file during a put operation.)

Class of Service is used as a means for specifying the amount of parallelism or stripe width for a file. See your HPSS system administrator for the Class of Service identifiers defined for your site. If a Class of Service is not specified, a default is used.

In the example below, the following commands might be entered to put a large file to HPSS with a Class of Service identifier of 4. In this example, 4 might designate 4-way striping to 3490 tape.

```
site setcos 4
```

2.1.2. Changing a File's Group by ID - chgid

chgid is used to change the group ID of a file and has the following format:

```
site chgid gid file
```

where,

gid is the new group ID of the file

file is the name of the file.

The user must belong to the specified group and be the owner of the file, or be the root user.

Example: The following may be entered to change the group ID of myfile to group ID 210.

site chgid 210 myfile

2.1.3. Changing a File's Group by Name - chgrp

chgrp is used to change the group name of a file and has the following format:

site chgrp *group file*

where,

group is the new group name of the file, and *file* is the name of the file.

The user must belong to the specified group and be the owner of the file, or be the root user.

Example: The following may be entered to change the group of myfile to group mygroup.

```
site chgrp mygroup myfile
```

2.1.4. Changing a File's Permissions - chmod

chmod is used to change the mode of a file and has the following format:

site chmod *mode file*

where,

mode is the new octal mode number of the file

file is the name of the file.

Mode is constructed from the OR of the following modes:

0400	read by owner
0200	write by owner
0100	execute (search in a directory) by owner
0040	read by group
0020	write by group
0010	execute (search in a directory) by group
0004	read by others
0002	write by others
0001	execute (search in a directory) by others

Note: The following mode values are not supported:

4000	set user ID on execution
2000	set group ID on execution
1000	sticky bit

Only the owner of the file or root user can change its mode.

Example: The following may be entered to change the mode of myfile to read, write by owner and group.

```
site chmod 0660 myfile
```

2.1.5. Changing a File's Owner by Name - chown

chown is used to change the owner of a file and has the following format:

site chown *owner file*

where,

owner is the new owner of the file

file is the name of the file.

Only the root user can change the owner of a file.

Example: The following may be entered to change the owner of /home/smith/myfile to jones.

```
site chown jones /home/smith/myfile
```

2.1.6. Changing a File's Owner by ID - chuid

chuid is used to change the uid of a file and has the following format:

site chuid *uid file*

where,

uid is the new uid of the owner of the file

file is the name of the file.

Only the root user can change the uid of a file.

Example: The following may be entered to change the uid of /home/smith/myfile to 201.

```
site chuid 201 /home/smith/myfile
```

2.1.7. Staging a File - stage

stage is used to initiate a stage of a migrated file (e.g. from tape to disk). The user can initiate the stage and then return at a later time to initiate the file transfer usin the FTP **get** or PFTP **pget** commands:

site stage *file*

where,

file is the name of the file.

Example: The following may be entered to stage file /home/smith/myfile.

```
site stage /home/smith/myfile
```

2.1.8. Setting the Desire Wait Options (for Migrated Files) - wait

wait is used to notify the HPSS PFTP Daemon :

site wait *option*

where,

option is one of the following values:

-1 or inf(inite) - wait forever for the file to be staged. Do not return from the get / pget command to complete until the file has been transferred or a transfer error has occurred.

0 - do not wait for the file to be staged. If the file has been migrated, return the appropriate message and initiate the stage. The user will return later to reissue the **get** / **pget** command.

n (where n is an integer) - wait the specified period (in seconds) for the file requested by a get / pget command to complete. Either transfer the file if the file is staged within the specified period or return a reply to notify the user to try again later.

Example: The following may be entered to wait for fies to be staged.

```
site wait -1
```

The following table describes the behaviour the customer should expect from FTP when issuing the stage/wait commands. Note: ONLY Classes of service utilizing the “Stage on Background” will exhibit predictable results.

Stage/Wait Behaviour

Wait Time	File Condition	Command	Behaviour /Message
No Wait	Archived	site stage xyz	"File xyz is being retrieved from archive."
No Wait	Not Archived	site stage xyz	"File xyz is currently ready for other processing."
Wait ###	Archived	site stage xyz	Wait for period then receive message: "File xyz is currently ready for other processing." or "File xyz is currently ready for other processing." if the file is staged in the time frame allowed
Wait ###	Not Archived	site stage xyz	"File xyz is currently ready for other processing."
No Wait	Archived	get xyz	"File xyz is being retrieved from archive."
No Wait	Not Archived	get xyz	Transfers Data as expected.
Wait ###	Archived	get xyz	Wait for period then receive message: "File xyz is being retrieved from archive." or transfers data as expected if file is staged in the time allowed.
Wait ###	Not Archived	get xyz	Transfers file as expected.

2.1.9. Creating a Symbolic Link - symlink

symlink is used to create a symbolic link.

site symlink *path/file link*

where,

path/file refers to the destination

link refers to the local filename.

Example: The following may be entered to create a link names sys_passwd in the local directory pointing to /etc/passwd.

```
site symlink /etc/passwd sys_passwd
```

A **dir** command will show sys_passwd -> /etc/passwd.

2.1.10. Allocating space for files - quote allo64

The quote allo64 command is used to specify the size of a file for space allocation.

quote allo64 *size*

where,

size is a string representing the size of the file. The size may be a decimal number less than 2^{64} or may be in the form 1MB (1048576). No spaces are allowed between the decimal number and the magnitude representation string. Accepted magnitude representation strings are:

KB (kilobyte = 1024),

MB (megabyte = 1048576),

GB (gigabyte = 1073741824),

TB (terabyte = 1099511627776),

PB (petabyte = 1125899906842624).

The magnitude representation string is case independent. The decimal component may contain up to two decimal points of precision. NOTE: 1005.03 will truncate to 1005 if no magnitude representation string is specified. Similar truncations will occur for excess precision specifications.

This command provides a 64-bit extension to the standard **quote allo** *size* command. NOTE: the **quote allo** *size* command only accepts decimal values for size. Both these commands are helpful for providing hints for non-parallel "put" commands.

Example: The following may be entered to specify the file size of 8 gigabytes.

```
quote allo64 8GB
```

2.2. List Directory Extensions

FTP supports the **ls** command to list the contents of a directory. Standard options supported are: **ls**, **ls -l**, **ls -a**, and **ls -F**. In addition to the standard **ls** options generally provided, HPSS also provides a **-lh** option. If **-lh** is specified, then a long directory listing is generated. However, in place of the owner field (field #3) and group field (field #4) listed for the -l option, the Class of Service identifier and Account Code are listed.

Example: **ls -lh**

```
-rw-rw---- 1 1      198   157286400 May 13 1996  TEST
-rw-r--r--  1 1      160       32768 May 16 1996  prod1
-rw-r--r--  1 1      160       32768 May 16 1996  prod10
-rw-r--r--  1 1      160       32768 May 16 1996  prod11
-rw-r--r--  1 1      160       32768 May 16 1996  prod12
-rw-r--r--  1 1      160       32768 May 16 1996  prod13
-rw-r--r--  1 1      160       32768 May 16 1996  prod14
-rw-r--r--  1 1      160       32768 May 16 1996  prod15
-rw-r--r--  1 1      160       32768 May 16 1996  prod151
-rw-r--r--  1 1      160       32768 May 16 1996  prod152
-rw-r--r--  1 1      160       32768 May 16 1996  prod153
-rw-r--r--  1 1      160       32768 May 16 1996  prod154
-rw-r--r--  1 1      160       32768 May 16 1996  prod155
-rw-r--r--  1 1      160       32768 May 16 1996  prod156
-rw-r--r--  1 1      160       32768 May 16 1996  prod157
-rw-r--r--  1 1      160       32768 May 16 1996  prod158
-rw-r--r--  1 1      160       32768 May 16 1996  prod159
```

Chapter 3. Parallel File Transfer Protocol (PFTP)

This chapter specifies the HPSS PFTP interface. For information on configuring the HPSS PFTP daemon, reference the *HPSS Administration Guide*. In order to use PFTP, the PFTP client code must be supported from the client platform.

PFTP supports the FTP command set plus some additional commands (refer to the next subsection). To use PFTP, the user enters one of the following commands:

```
pftp_client [-g] [-i] [-m[filename] [-n] [-v] [-w###] [Host [Port]]
pftp_client_ipi3 [-g] [-i] [-m[filename] [-n] [-v] [-w###] [Host [Port]]
```

where,

-g	Disables the expansion of metacharacters in file names. Interpreting metacharacters can be referred to as expanding (sometimes called globbing) a file name. See the glob subcommand.
-i	Turns off interactive prompting during multiple file transfers. See the prompt, mget, mput, and mdelete subcommands for descriptions of prompting during multiple file transfers.
-m	This argument will enable multinode processing. By default, multinode processing is disabled. The full path and filename of the multinode configuration file can be specified immediately following this argument
-n	Prevents an automatic login on the initial connection. Otherwise, the ftp command searches for a $HOME/.netrc entry that describes the login and initialization process for the remote host. See the user subcommand.
-v	Displays all the responses from the remote server and provides data transfer statistics. This display mode is the default when the output of the ftp command is to a terminal, such as the console or a display.
-w	This argument will set the pwidth. The pwidth value must be specified immediately following this argument.
Host	The node where the HPSS FTP Daemon process resides.
Port	`PortThe port number for the HPSS ftpd, as set up in /etc/services.`

The **pftp_client_ipi3** command would be use when file transfers use the IPI-3 protocol. The local administrator may opt to define a **pftp** program link that points to either **pftp_client** or **pftp_client_ipi3**, as appropriate.

Two additional forms of the **pftp_client** are available (upon request and pursuant to legal restrictions): **krb5_gss_pftp_client** and **dce_gss_pftp_client**. Since the Generic Security Service (GSS) versions of the Parallel FTP Client only relate to the authentication process, the HIPPI (IPI3) forms of these applications should behave the same as the non-GSS versions. The GSS clients are used to provide credential authentication facilities (password-less authentication) between the client and the HPSS GSS Parallel FTP Daemon using either Kerberos or DCE credentials for authentication.

The GSS versions of the Parallel FTP Client require either Kerberos and/or DCE Client software and the Data Encryption Standard (DES) libraries. The Kerberos version is based on MIT Kerberos Beta Version 5.6. Updates are anticipated in the future; however, neither IBM nor the HPSS Development team are obligated to pursue this path.

As a courtesy to HPSS customers, the Parallel FTP Client code is available for compilation at customer sites upon request. The Parallel FTP Client code will be provided as a .tar.Z file (tarred and compressed) containing all components required to build the applications. Hardware/Software dependencies are the individual HPSS customers responsibility. This explicitly denies any support requirement on IBM or the HPSS Development/Support personnel for any modifications made by the customer. No DCE software is required to build the HPSS Parallel FTP Client.

The HPSS PFTP Client has been successfully compiled on: Cray UNICOS, Hewlett-Packard HPUX, Silicon Graphics IRIX (32 Bit), Sun Solaris, Intel Paragon OSF, Intel Teraflop OSF, Linux Intel, and IBM AIX 4.x. Ports to other hardware/software components are the responsibility of the remote site. These sites will be asked to share their ports with the HPSS development team (and other HPSS facilities); however, neither IBM nor the HPSS Development Team accepts any obligation to incorporate any hardware/software ports into the distribution source. No site specific features (local mods) added to the Parallel FTP client by customer sites will be incorporated into the PFTP client without the modification of the HPSS license.

The GSS versions of the Parallel FTP Client require MIT Kerberos and/or the DCE Toolkit. The Kerberos V5B6 code is available upon request (pursuant to restrictions). Neither IBM nor the HPSS Development team declare the Kerberos code suitable for any specific purpose nor are they obligated to repair or support customers using this code. The GSS PFTP Client code has been compiled on each of the above except Linux Intel.

The GSS HPSS Parallel FTP Daemon is available for IBM AIX 4.x only.

Note: If the message "Load thread state failed" is received, contact your HPSS administrator. This message generally implies that either HPSS is not correctly configured, or some HPSS components may not be executing.

3.1. Parallel FTP Client Transfers

When the pwidth value is set and a valid multinode configuration file does not exist, the default behaviour of PFTP is to provide parallel data paths to the client by spawning multiple processes on the client node. A multinode option supports spawning the client processes across multiple nodes. This multinode option would be beneficial on processors which support shared file systems, such as GPFS on the IBM SP. Note: without a shared file system, a multinode file transfer to the client would be spread across multiple, separate files, which may not be the desired behaviour. The client nodes which participate in a multinode transfer are selected from a configuration file which contains entries with control and optionally data interface names or addresses. The number of nodes selected from the configuration file is based on the pwidth value. The starting node is selected using an offset of which is maintained by the PFTP client. The offset is automatically updated as transfers are performed in order to better load balance across the client nodes.

3.1.1. Parallel FTP Client Configuration

The PFTP client will need to be configured to support a multinode parallel transfer.

3.1.1.1. Multinode Configuration File

A configuration file is used to control specifying different IP address of remote nodes on which the multinode daemons are to be spawned by the PFTP client. The file contains the following information:

Control Interface
The IP address or workstation name where a multinode daemon has been configured to run.

Data Interface
The IP address or workstation name to be used by the multinode daemon to move data between the mover to the multinode daemon. If a data interface is not specified, the control interface will be used as the data interface.
Each instance of the PFTP client builds its own internal node list using the configuration file. The PFTP client will use the same internal node list, to transfer files. Nodes can only be added to this list by increasing the pwidth. Decreasing the pwidth will not remove nodes from the internal node list. Because of this, it is very important to properly order the configuration file by alternating the interfaces of various processors.

Sample entries in a configuration file:

```
# Control_Interface Data_Interface Comment
192.94.47.21                 # pluto:   control and data share 192.94.47.21
192.94.47.43                 # hpss :   control and data share 192.94.47.43
pluto                        # pluto:   control and data share pluto (192.94.47.21)
hpss                         # hpss :   control and data share hpss (192.94.47.43)
192.94.47.21 192.225.22.21   # pluto:   control on 192.94.47.21
                             #          data on 192.225.22.21
192.94.47.43 192.225.22.43   # hpss :   control on 192.94.47.43
                             #          data on 192.225.22.43
pluto pluto-F                # pluto:   control on pluto
                             #          data on pluto-F
```

3.1.1.2. Multinode Load Leveling:

Load leveling across the various interfaces is accomplished using an offset file and a properly "ordered" configuration file. The offset file is automatically created and maintained by the PFTP client software. The offset file will have the same path and name as the configuration file, with the ".off" extension. The user must have both read and write permissions to the offset file. The offset file is used to level the load across every node in the configuration file, for each instance of the PFTP client. By placing the configuration file in a shared location, network wide load leveling can be achieved.

3.2. Additional Commands

All FTP extensions described in Chapter 5 are supported by PFTP. In addition, the following commands are supported by PFTP.

- **pappend**
- **pput, mpput**
- **pget, mpget**
- **psocket**
- **pipi3**
- **setpwidth**
- **setpblocksize**
- **multinode**

3.2.1. Parallel append - pappend

Synopsis

pappend *local_file* [*remote_file*]

Description

The **pappend** command transfers a file from the local machine to HPSS. The transfer starts at the end of the remote file and continues until the entire file is moved or until an error occurs.

Parameters

local_file	Identification of the file to transfer on the local machine.
remote_file	Optional file name to the remote file. If not supplied then the remote (HPSS) file name defaults to be the same as the local file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures: data transfer connection malfunction.

Network Failures: data transfer malfunction.

Allocation Failures: no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

-5 - an I/O error occurred.

-28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Append local file testfile to the same file name in the user's HPSS home directory.

```
pappend testfile
```

2. Append local file testfile to HPSS file prod1 in the current working directory.

```
pappend testfile prod1
```

3.2.2. Parallel file store - pput

Synopsis

pput [-l *local_offset*] [-r *remote_offset*] [-s *size*] *local_file* [*remote_file*]

Description

The **pput** command transfers a file from the local machine to HPSS. If offsets and size of transfer are not specified, the transfer starts at the beginning of the local file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying local file offset, remote file offset, and size of transfer. The *local_offset*, *remote_offset*, and *size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation.

The normal **pput** command functions just like the standard ftp **put** command and transfers an entire file.

Parameters

-l *local_offset*	Optional byte offset into the local file where the transfer is to begin.
-r *remote_offset*	Optional byte offset into the remote file where the data is to be placed.
-s *size*	Optional byte size of the amount of data to transfer.
local_file	Identification of the file to transfer on the local machine.
remote_file	Optional file name to the remote (HPSS) file. If not supplied then the remote file name defaults to be the same as the local file name.

Return strings

Output shows amount of data transferred and any error conditions.

Error conditions

Connection Failures:	data transfer connection malfunction.
Network Failures:	data transfer malfunction.

Allocation Failures: no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

-5 - an I/O error occurred.

-28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer local file testfile to the user's HPSS home directory.

```
pput testfile
```

2. Transfer local file testfile to HPSS file prod1 in the current working directory.

```
pput testfile prod1
```

3. Transfer 1MB from offset 1MB of local file testfile to offset 0 of HPSS file /home/bob/prod1.

```
pput -l 1048576 -r 0 -s 1048576 testfile /home/bob/prod1.
```

3.2.3. Parallel file store - mpput

Synopsis

mpput *local_files*

Description

The **mpput** command expands the files specified in the *local_files* parameter at the local host and copies the indicated files to HPSS. The **mpput** command functions just like the standard ftp **mput** command.

Parameters

local_files	Identification of the files to transfer on the local machine.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures: data transfer connection malfunction.

Network Failures: data transfer malfunction.

Allocation Failures: no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

-5 - an I/O error occurred.

-28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all local files in the current directory to the user's HPSS home directory.

```
mpput *
```

2. Transfer all local files which begin with test in directory /usr/bob to the user's HPSS home directory.

```
mpput /usr/bob/test*
```

3.2.4. Parallel file retrieval - pget

Synopsis

```
pget [-r remote_offset] [-l local_offset] [-s size] remote_file
[local_file]
```

Description

The **pget** command transfers a file to the local machine from HPSS. If offsets and size of transfer are not specified, the transfer starts at the beginning of the remote file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying remote file offset, local file offset, and size of transfer. The *local_offset*, *remote_offset*, and *size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation.

The standard **pget** command transfers entire files similar to the standard **ftp** get command.

Parameters

-r *remote_offset*	Optional byte offset where transfer is to begin in the remote file.
-l *local_offset*	Optional parameter where the data is transferred in the local file.
-s *size*	Optional number of bytes to transfer.
remote_file	Identification of the file to transfer from the remote (HPSS) host.
local_file	Optional file name to the local file. If not supplied then the local file name defaults to be the same as the remote file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures:	data transfer connection malfunction.
Network Failures:	data transfer malfunction.
Allocation Failures:	no space on local machine for file.

Error codes may also be returned from HPSS. The most common error code is:

-5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer HPSS file /home/bob/prod1 to the user's local directory.
   `pget /home/bob/prod1`

2. Transfer HPSS file prod1 in the current working directory to local file testfile1.
   pget prod1 testfile1

3. Transfer 1MB from offset 0 of HPSS file /home/bob/prod1 to offset 1048576 of local file testfile.
   pget -r 0 -l 1048576 -s 1048576 /home/bob/testfile1 testfile

3.2.5. Parallel file retrieval - mpget

Synopsis

mpput *remote_files*

Description

The *mpget* command expands the *remote_files* parameter at the remote (HPSS) host and copies the indicated HPSS files to the current directory on the local host. The **mpget** command functions just like the standard ftp **mget** command.

Parameters

remote_files	Identification of the files to transfer from the remote (HPSS) host.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures: data transfer connection malfunction.

Network Failures: data transfer malfunction.

Allocation Failures: no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error code is:

-5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all files in HPSS directory /home/bob to the user's local directory.

```
mpget /home/bob/*
```

2. Transfer all HPSS files which begin with test in directory /home/bob to the user's local directory.

```
mpget /home/bob/test*
```

3.2.6. Specify TCP socket based transfers - psocket

Synopsis

```
psocket
```

Description

The **psocket** command is used to specify to the FTP client code that any parallel transfers are now to be done using connection based sockets (TCP).

Parameters

none.

Return strings

"Parallel transfers will now go over sockets."

Error conditions

none.

See also

RFC-0959.

Notes

If neither **psocket** nor **pipi3** is specified, **psocket** is the default.

3.2.7. Specify IPI-3 based transfers - pipi3

Synopsis

```
pipi3
```

Description

The **pipi3** command is used to specify to the FTP client code that any parallel transfers are now to be done using the IPI-3 protocol. This protcol allows the PFTP client to transfer data directly to or from a HIPPI-attached device.

Parameters

none.

Return strings

"Parallel transfers will now go over ipi3."

Error conditions

If IPI-3 is not supported then the following will be printed:

"IPI-3 transfers not allowed"

See also

RFC-0959.

Notes

This option is only appropriate for the **pftp_client_ipi3** command.

If neither **psocket** nor **pipi3** is specified, **psocket** is the default.

3.2.8. Specify transfer stripe width - setpwidth

Synopsis

```
setpwidth stripe_width
```

Description

The **setpwidth** command is used to specify the size of the client side stripe to the FTP client code.

Parameters

stripe_width

The width of the PFTP client-side stripe. The width can have a value of 1 through 16. The default width is 1. The stripe width from the PFTP client perspective is the number of client processes spawned to handle the data transfers. Stripe width from the server perspective is the number of volumes the file is striped across.

A general guideline would be to set *stripe_width* to an even divisor of the number of volumes the file is striped across. For example, if the Class of Service for a file were set up for a 4-way stripe, suggested values for *stripe_width* might be 2 or 4.

If the stripe width of the file is unknown, consult your HPSS administrator or follow the following steps to determine the stripe width.

1. Enter the **lshpss -cos** command to list Class of Service information. From the entry with the *CID* value equal to your Class of Service ID, locate the *HID* (hierarchy ID) field.

2. Enter the **lshpss -hier** command to list hierarchy information. From the entry matching the *HID* value above, locate the *StorageClassID* field.

3. Enter the **lshpss -sc** command to list the storage class information. From the entry matching the *StorageClassID* value above, locate the *W* field. This is the stripe width for the storage class of the file.

Return strings

"Parallel stripe width set to [stripe width]."

Error conditions

"Bad width value [stripe width]."

See also

RFC-0959.

Notes

none.

Example

1. Set the stripe width to 4.
   setpwidth 4

3.2.9. Specify transfer block size - setpblocksize

Synopsis

setpblocksize *block_size*

Description

The **setpblocksize** command is used to specify the block size to be used for parallel transfers. The *block_size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation. The maximum blocksize is 16mb.

Parameters

block_size — The number of bytes to be transferred to each element of the stripe before data is sent to the next element. The current allowable transfer sizes range from 1 through 16MB. The default block size is 256KB

A general guideline would be to set *block_size* to the virtual volume block size. Consult your HPSS administrator or follow the following steps to determine the virtual volume blocksize.

1. Enter the **lshpss -cos** command to list Class of Service information. From the entry with the *CID* value equal to your Class of Service ID, locate the *HID* (hierarchy ID) field.
2. Enter the **lshpss -hier** command to list hierarchy information. From the entry matching the *HID* value above, locate the *StorageClassID* field.
3. Enter the **lshpss -sc** command to list the storage class information. From the entry matching the *StorageClassID* value above, locate the *VVBlk* field. This is the virtual volume block size.

Return strings

"Parallel block size set to [block size]."

Error conditions

"Bad block size value [block size]."

See also

RFC-0959.

Notes

none.

Example

1. Set the transfer block size to 8 MB.

```
setpblocksize 8388608

setpblocksize 8MB
```

3.2.10. Multinode Enable/Disable - multinode

Synopsis

multinode

Description

The **multinode** command is used to enable/disable the "desire" to perform a parallel file transfer using multiple nodes. When multinode is enabled, the **pftp_client** will process the multinode configuration file. If the process cannot obtain a single node to perform the parallel transfer, then the transfer will occur using non-multinode parallel method.

Parameters

None.

Return strings

“Processing the multinode list, please wait.....”

“Multinode is on.”

or

“Multinode is off.”

Error conditions

“Configuration File I/O Problems: Without nodes, files cannot be transferred using the multiple node capability.”

See also

None.

Chapter 4. User Utilities

HPSS provides a set of utilities for administrators and users. The majority of the HPSS utilities are for administrators, and are defined in the HPSS Administration Guide. Those utilities applicable to users are documented in this chapter. There are also man pages for these utilities. Note: The user issue a DCE login prior to using the utilities.

4.1. Utilities

The user utilities defined in the chapter are:

chacl
lsacl
lshpss

4.1.1. Change ACL utility - chacl

Synopsis

chacl [-r -u -b] < *file*

Description

chacl is used to change discretionary access control information associated with a specified object. The output from **lsacl** is in the correct format for input to *chacl*. The input to chacl for an ACL consists of two sections: the comments section and the ACL entries section. The comments section contains three lines: name of the object, the object owner, and the object owning group. The ACL entries section contains lines with three fields separated by a colon. The first field contains the ACL entry tag type. The second field contains the ACL entry qualifier. The third field contains the access permissions.

Output Format:

```
#file:<filename>
#owner:<uid>
#group:<gid>
```

```
user::<perm>
mask::<perm>
user:<uid>:<perm>
user:<uid2>:<perm>
group::<perm>
group:<gid>:<perm>
group:<gid2>:<perm>
other::<perm>
```

The types of permissions are:

```
r - read
w - write
x - execute
i - insert
d - delete
a - append
m - modify
```

The utility **chacl** is initiated from the command line. The utility differs from some other HPSS utilities in that it uses the Client API library. The user must have issued a **dce_login**. If HPSS was configured without the default values, the HPSS_LS_NAME environment variable nust be set to the CDS name of the Location Server DCE group.

Parameters

-r	Remove the specified ACL entries. Base entries are not affected.
-u	Update the specified ACL entries. The entries are added if they do not already exist.
-b	Remove all entries except the three base entries, user::, group::, and other::.
file	The file containing the object to change.

See also

lsacl

Notes

none

Example

The following steps add the user with ID 7004 with read permission to the ACL list of the file /ampex/test.txt.

```
dce_login hpss_client1

export HPSS_LS_NAME=/.:/hpss/ls/group

lsacl -f /ampex/test.txt > acl_format

cat acl_format
#file: /ampex/test.txt
#owner:1006
#group:system
user::rwx---m
mask::r-x---
group::r-x---
```

Now edit acl_format to add the line "user:7004:r-----"

```
cat acl_format
#file: /ampex/test.txt
#owner:1006
#group:system
user::rwx---m
user:7004:r------
mask::r-x---
group::r-x---

chacl -u < acl_format
```

4.1.2. List ACL utility - lsacl

Synopsis

lsacl [**-f** *object_path_name*]

Description

lsacl is used to list access control information associated with a specific path name. The output from **lsacl** is in the correct format for input to chacl(change ACL). The output from **lsacl** for an object ACL consists of two sections: the comments section and the ACL entries section. The comments section contains three lines: name of the object, the object owner, and the object owning group. The ACL entries section contains lines with three fields separated by a colon. The first field contains the ACL entry tag type. The second field contains the ACL entry qualifier. The third field contains the access permissions.

Output Format:

```
#file:<filename>
#owner:<uid>
#group:<gid>
user::<perm>
mask::<perm>
user:<uid>:<perm>
user:<uid2>:<perm>
group::<perm>
group:<gid>:<perm>
group:<gid2>:<perm>
other::<perm>
```

The types of permissions are:

```
r - read
w - write
x - execute
i - insert
d - delete
a - append
m - modify
```

The utility **lsacl** is initiated from the command line. The path name is the key input value. The utility differs from some other HPSS utilities in that it uses the Client API library. The user must have issued a **dce_login**. If HPSS was configured without the default values, the HPSS_LS_NAME environment vari-

able nust be set to the CDS name of the Location Server DCE group.

Parameters

-f *object_path_name* The HPSS path name of the object whose ACLs are to be listed.

See also

chacl

Notes

none.

Example

The following steps list the ACLs associated with the directory /ampex owned by the HPSS user hpss_client1.

```
dce_login hpss_client1

export HPSS_LS_NAME=/.:/hpss/ls/group

lsacl -f /ampex         {input}
#file: /ampex           {output}
#owner:1006
#group:system
user::rwx---m
mask::r-x---
group::r-x---
```

4.1.3. List information about HPSS - lshpss

Synopsis

lshpss [**options ...**]

Description

lshpss displays HPSS resources, such as Class of Service, Hierarchy, and Storage Class. Before running this script, you must be authorized to access the SFS files. This is accomplished by entering **dce_login** for an authorized DCE ID.

Parameters

-s *sfs_server*	Specify optional Encina SFS Server. The SFS Server can also be set by setting the **ENCINA_SFS_SERVER** environment variable (e.g. export **ENCINA_SFS_SERVER**=/.:/encina/sfs/hpss)
-cos	Show Class of Service list
-hier	Show Hierarchy list
-sc	Show Storage Class list
-migp	Show Migration Policies
-purgep	Show Purge Policies
-vol	Show Physical Volumes
-dev	Show Mover Devices
-drv	Show PVL Drives
-svr	Show HPSS Servers
-mvr	Show HPSS Movers
-acct	Show Accounting Policies
-logp	Show Log Policies
-locp	Show Location Policies

-ffam	Show file families
-bfs	Show Bitfile Servers
-ns	Show Name Servers
-pvr	Show PVRs
-logd	Show Log Deamons
-logc	Show Log Clients
-nfsd	Show NFS Daemons
-listmeta	List metadata
-all	Display all of the above
-dumpmeta	Dump all metadata for HPSS to separate files
-dump	Dump one metadata file
-sdt	Go into an sdt shell
-h	Show this help message

See also

none.

Notes

The option of interest to most users is the -cos option. This option allows the user to view a list of all defined Classes of Service.

Example

The following steps list the Classes of Service, hierarchies, and storage classes:

```
dce_login hpss_client1

lshpss -cos -hier -sc
```

Sample output from this example is shown below:

```
                                - Class of Service List -
                           - Class of Service List -

                                        Optim                                  Xfer Avg
COS Hier                                Access            File Size            Rate Lat
ID  ID   Name                           Size              Min / Max            kB/s (s) SC Flags
--- ---- ------------------------------ ------ -------------------------------- ---- --- -- -----
  1    1 larry: very small files (H1)      4MB               0/    17179869184 4096   0  O RWAM-
  2    2 larry: small files/2 copy (H2)    4MB         4194304/       16777216 4096   0  O RWAM-
  3    3 larry: small files (H3)           4MB         4194304/       16777216 4096   0  O RWAM-
  4    4 larry: medium files (H4)          4MB        16777216/    68719476736 4096   0  O RWAM-
  5    5 larry: large files (H5)           4MB        67108864/   214748364800 4096   0  O RWAM-
  6    6 larry: 1w tape files (H6)         4MB               0/   107374182400 4096   0  O RWAM-
  7    7 larry: 2wt tape files (H7)        4MB               0/   107374182400 4096   0  O RWAM-
  8    3 larry: small files_b (H3)         4MB               4/       16777216 4096   0  A RWA--
--- ---- ------------------------------ ------ -------------------------------- ---- --- -- -----
SC=Stage Code (N=none, O=on open, A=async, B=background)

Flags: RWAMF
R=Enable read               W=Enable write        A=Enable append
M=Enforce max file size     F=Force selection


                      - Hierarchy List -

Hier                                    #
  ID Description                        Lev Storage Class IDs
---- ---------------------------------- --- --------------------------------
   1  larry: 1wd -> 1wt (H1)               2 1 --> 101
   2  larry: 1wd -> 1wt -> 1wt (H2)        3 2 --> 102 --> 103
   3  larry: 1wd->1wd->1wt->1wt (H3)       4 3 --> 4 --> 104 --> 105
   4  larry: 2wd -> 1wt (H4)               2 5 --> 106
   5  larry: 4wd -> 1wt (H5)               2 6 --> 107
   6  larry: 1wt -> 1wt (H6)               2 108 --> 109
   7  larry: 2wt (H7)                      1 110
---- ---------------------------------- --- --------------------------------


                                - Disk Storage Class List -

                                        Xfer      SSeg          Est             Media Thresh  Mig Prg
 SC                              Media    Rate      Size    Avg # PV          Str Block (% use) Pol Pol
 ID Name                          Type    (kB/s)   min/max   SSegs Size   VVBS  Wid Size  Wrn/Crt ID  ID
--- ---------------------------- -------- ------ ----------- ----- ------ ----- --- ----- ------- --- ---
                           ---
  1 larry: 1wd (H1-L1)            Default    3072  256K/  1MB     4    4GB  256K   1    4K  80/ 90   1   1
  2 larry: 1wd (H2-L1)            Default    3072   2MB/  4MB     4    4GB   1MB   1    4K  80/ 90   2   1
  3 larry: 1wd (H3-L1)            Default    3072   2MB/  4MB     4    4GB   1MB   1    4K  80/ 90   0   0
  4 larry: 1wd (H3-L2)            Default    3072   2MB/  4MB     4    4GB   1MB   1    4K  80/ 90   1   1
  5 larry: 2wd (H4-L1)            Default   12288   8MB/ 16MB     4    4GB   1MB   2    4K  80/ 90   1   1
  6 larry: 4wd (H5-L1)            Default   49152  32MB/256MB     4    4GB   1MB   4    4K  80/ 90   1   1
 --- ---------------------------- -------- ------ ----------- ----- ------ ----- --- ----- ------- -- --
VVBS=Virtual volume block size
```

- Tape Storage Class List -

SC ID	Name	Media Type	Media Block Size	Est PV Size	Str Wid	VVBS	Xfer Rate (kB/s)	BBTM	Threshold (volumes) Warn/Crit	Max VVs to Write	Mig Pol ID	Prg Pol ID
101	larry: 1wt 3590E (H1-L2)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
102	larry: 1wt 3590E (H2-L2)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
103	larry: 1wt 3590E (H2-L3/2nd cp	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
104	larry: 1wt 3590E (H3-L3)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	1	0
105	larry: 1wt 3590E (H3-L4)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
106	larry: 1wt 3590E (H4-L2)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
107	larry: 1wt 3590E (H5-L2)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
108	larry: 1wt 3590E (H6-L1)	3590	256K	40GB	1	1MB	14336	280	2/ 1	1	1	0
109	larry: 1wt 3590E (H6-L2)	3590	256K	40GB	1	1MB	14336	280	2/ 1	2	0	0
110	larry: 2wt 3590E (H7-L1)	3590	256K	40GB	2	1MB	57344	560	2/ 1	2	0	0

BBTM=Blocks between tape marks
VVBS=Virtual volume block size

Appendix A - Acronyms

ACL	Access Control List
ACLS	Automated Cartridge System Library Software (Science Technology Corporation)
AIX	Advanced Interactive Executive
API	Application Program Interface
CDS	Cell Directory Server
COS	Class of Service
DCE	Distributed Computing Environment
DFS	Distributed File System
EFS	External File System
FTP	File Transfer Protocol
gid	Group Identifier
GSS	Generic Security Service
HIPPI	High Performance Parallel Interface
HPSS	High Performance Storage System
IBM	International Business Machines Corporation
LaRC	Langley Research Center
LANL	Los Alamos National Laboratory
LLNL	Lawrence Livermore National Laboratory
NASA	National Aeronautics and Space Administration
ORNL	Oak Ridge National Laboratory
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input / Output
IP	Internet Protocol
IPI	Intelligent Peripheral Interface
NFS	Network File System
OSF	Open Software Foundation
PFTP	Parallel File Transfer Protocol
RISC	Reduced Instruction Set Computer
SFS	Structured File Server
SNL	Sandia National Laboratories
SP	Scalable Processor
TCP	Transmission Control Protocol
uid	User Identifier
VV	Virtual Volume

Appendix B - References

1. *File Transfer Protocol, RFC-0959,* October 1985.

2. *HPSS Error Messages Manual*, November 1999.

3. *HPSS Programmer's Guide Reference, Volume 1*, November 1999.

4. *HPSS Programmer's Guide Reference, Volume 2*, November 1999.

5. *HPSS System Administration Guide*, November 1999.

6. *Installing, Managing, and Using the IBM AIX Parallel I/O File System*, Document Number H34-6065-00.

7. *Network File System Specification, RFC-1094*, DDN Network Information Center, SRI International, Menlo Park, Ca.

8. *OSF DCE User's Guide and Reference*, Prentice Hall, Englewood Cliffs, N. J.

Index